Disguised Executable Files in Spear-Phishing Emails: Detecting the Point of Entry in Advanced Persistent Threat

Ibrahim Ghafir[1,2], Vaclav Prenosil[1], Mohammad Hammoudeh[3], Francisco J. Aparicio-Navarro[4], Umar Raza[3], Ahmad Jabban[5]

[1] Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2] Department of Computer Science, Durham University, Durham, UK
[3] Faculty of Science & Engineering, Manchester Metropolitan University, Manchester, UK
[4] School of Engineering, Newcastle University, Newcastle upon Tyne, UK
[5] Univ Rennes, INSA Rennes, Rennes, France
{ghafir,prenosil}@fi.muni.cz,{m.hammoudeh,u.raza}@mmu.ac.uk,francisco.aparicio-navarro@ncl.ac.uk

ABSTRACT

In recent years, cyber attacks have caused substantial financial losses and been able to stop fundamental public services. Among the serious attacks, Advanced Persistent Threat (APT) has emerged as a big challenge to the cyber security hitting selected companies and organisations. The main objectives of APT are data exfiltration and intelligence appropriation. As part of the APT life cycle, an attacker creates a Point of Entry (PoE) to the target network. This is usually achieved by installing malware on the targeted machine to leave a back-door open for future access. A common technique employed to breach into the network, which involves the use of social engineering, is the spear phishing email. These phishing emails may contain disguised executable files. This paper presents the disguised executable file detection (DeFD) module, which aims at detecting disguised exe files transferred over the network connections. The detection is based on a comparison between the *MIME type* of the transferred file and the file name extension. This module was experimentally evaluated and the results show a successful detection of disguised executable files.

CCS CONCEPTS

•**Security and privacy** → **Intrusion detection systems;** *Network security;* •**Networks** → Network monitoring;

KEYWORDS

Cyber attacks, advanced persistent threat, spear-phishing emails, disguised executable file, malware, intrusion detection system.

ACM Reference format:
Ibrahim Ghafir[1,2], Vaclav Prenosil[1], Mohammad Hammoudeh[3], Francisco J. Aparicio-Navarro[4], Umar Raza[3], Ahmad Jabban[5]. 2018. Disguised Executable Files in Spear-Phishing Emails: Detecting the Point of Entry in Advanced Persistent Threat. In *Proceedings of ICFNDS '18, Amman, Jordan, June 26-27, 2018,* 5 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

ICFNDS '18, Amman, Jordan

DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Cyber attacks have become more widespread and several attacks have made headline news over the last 5 years, targeting industrial companies and governmental organisations [1] [2]. These attacks have caused substantial financial losses and been able to stop fundamental public services. The annual cost of cybercrime was $3 trillion in 2015. Furthermore, it is expected to increase to more than $6 trillion per annum by 2021 [3]. The term cyber attack generally refers to criminal activities lunched via the Internet, aiming usually at financial gains or stealing confidential data. Additionally, the attackers can spy and monitor the target organisation and disturb its functions, motivated by a political, ideological or criminal goal [4].

These challenges have brought much interest in the research and investment towards developing new tools and approaches for cybersecurity. In recent years, Advanced Persistent Threats (APTs) [5] have emerged, hitting selected companies and organisations. The main objectives of APT are data exfiltration and intelligence appropriation. This attack targets selected organisations and persists on the completion of the attack until it is achieved. The economic damage due a successful APT can be very expensive. Furthermore, the attacker can compromise the target network for a long period of time without being detected. This type of attack is currently one of the most serious threat to the cyber security [6].

As part of the APT life cycle, an attacker creates a Point of Entry (PoE) to the target network. This is usually achieved by installing malware on the targeted machine to leave a back-door open for future access. A common technique employed to breach into the network, which involves the use of social engineering, is the spear phishing email. These emails, which may contain disguised exe files, continue to be a favoured means by APT attackers to infiltrate target networks [7]. In a typical spear-phishing attack, a specially crafted email is sent to specific individuals from a target organization. The recipients are convinced through clever and relevant social engineering tactics to either download a malicious file attachment or to click a link to a malware or an exploited site, starting a compromise [8]. The malicious link may lead to a drive-by download attack, in which the user's computer can be infected with malware by only visiting a web site that contains the malicious content. Then, the malicious code which is installed on the victim's machine can control the infected machine and perform malicious activities [9]. Typically, sensitive data is exfiltrated, passwords are stolen and keystrokes are recorded.

In [10], we present a correlation-based system for APT detection, which runs through two main phases. First, eight detection modules are employed to detect the individual APT steps. These modules are Tor connection detection [11], malicious SSL certificate detection [12], malicious file hash detection [13], malicious domain name detection [14], domain flux detection [15], malicious IP address detection [16], scan detection and disguised exe file detection. Second, a correlation methodology including clustering algorithms is utilised to correlate the steps related to one APT campaign. This paper presents the disguised exe file detection (DeFD) module, which aims at detecting disguised exe files transferred over the network connections. The detection is based on a comparison between the *MIME type* of the transferred file and the file name extension.

The rest of this paper is structured as follows. Section 2 gives an overview of the APT life cycle. Current approaches for APT detection are described in Section 3. Section 4 presents the proposed methodology for the disguised exe file detection. Experimental evaluation is provided in Section 5 and Section 6 concludes the paper.

2 THE APT LIFE CYCLE

APT refers to Advanced Persistent Threat. APTs are a cybercrime category directed at business and political targets. APTs require a high degree of stealth over a long period of operation in order to be successful. The attacker usually aims for more than immediate financial gain, and infected systems continue to be compromised even after the target's network has been breached and initial goals reached [5]. Figure 1 depicts the steps of the APT attack [17].

Figure 1: Typical steps of the APT attack.

(1) Intelligence gathering: This initial step aims to get information regarding the target, like its organizational structure, IT environment and even about people who are working for that target. For this purpose, the attacker can use public sources (LinkedIn, Facebook, Twitter, etc) and prepare a customized attack. Through this step the attacker tries to find and organize accomplices, build or acquire tools, and research target/infrastructure/employees.

(2) Initial compromise (Point of entry): Performed by use of social engineering and spear phishing, over email, using software vulnerabilities [18]. Another popular infection method is planting malware on a website which the victim employees will be likely to visit.

(3) Command and control (C&C) communication: After an organization's perimeter has been breached, continuous communication between the infected host and the C&C server should be preserved to instruct and guide the compromised machine [19] [20]. These communications are usually protected by Secure Sockets Layer (SSL) encryption, making it difficult to identify if the traffic directed to sites is malicious. Another technique can be used in this step is domain flux technique [21]; an exploited host may try to connect to a large number of domain names which are expected to be C&C servers. The goal of this technique is to make it difficult or even impossible to shut down all of these domain names.

(4) Lateral movement: Once getting access to the target's network, the attacker laterally moves throughout the target's network searching for new hosts to infect. The attacker can use brute force attack [22], to obtain information such as a user password or personal identification number (PIN); an automated software is used to generate a large number of consecutive guesses as to the value of the desired data. Another technique is pass the hash attack [23], in which the attacker steals a hashed user credential and, without cracking it, reuses it to trick an authentication system into creating a new authenticated session on the same network.

(5) Asset/Data discovery: This step aims to identify and isolate the noteworthy assets within the target's network for future data exfiltration. Port scanning can be used for this step [24].

(6) Data exfiltration: Data of interest is transmitted into external servers which are controlled by the attacker. There are some techniques used for data exfiltration like built-in file transfer, via FTP or HTTP and via the Tor anonymity network [25].

3 RELATED WORK

The APT detection is a challenge for the current Intrusion Detection Systems (IDSs), and much research has been conducted to address this type of multi-stage attack. The authors in [26] propose an approach based on C&C domains detection. This approach processes the network traffic and depends on a conclusion that the connection to the C&C domain is independent, while the connection to the legitimate one is correlated. In [4], the authors introduce an active-learning-based system for APT detection. This system detects the malicious PDFs files which might be included in a phishing email or a malicious website.

The work in [27] describes a context-based system for targeted attacks detection. This system models APT as an attack pyramid, where the attack objective is on the top of the pyramid. Then, The environments, where the APT steps take place, form the lateral planes. The authors in [28] presents an APT detection system called TerminAPTor. This system tracks the information flow aiming to correlate meta alerts, generated for the individual APT steps, that may belong to one APT campaign. These meta alerts are fed to the system by an agent, which can be a standard IDS. The system explained in [29] is similar to TerminAPTor. However, the meta alerts are correlated based on a statistical methodology. The proposed framework describes APT as a multi-stage attack running through five stages which are delivery, exploit, installation, C&C and actions. Furthermore, it considers each stage to include several activities.

The authors in [30] explore Duqu, which is a piece of malware was used to launch an APT against a European company. The main objective of this APT campaign was data exfiltration. Additionally, this work presents a toolkit developed to detect the Duqu malware. This toolkit includes six investigation tools to analyse the traffic related to the Duqu malware. In [31], a framework to detect phishing emails is proposed. These phishing emails are utilised to initiate the point of entry for an APT. This framework performs a mathematical and computational analysis to cluster emails into malicious or benign ones. The developed algorithm makes use of "Tokens", which are patterns of words and characters such as (DHL, notification, delivery, label, invoice and post).

A machine learning approach is introduced in [32] for APT detection. The algorithm utilises four features extracted from the users' traffic which are the open ports, memory usage, CPU usage and number of files in the system32 folder. A normal traffic is used to build the normal baseline of the system. Then, a piece of malware, previously used in an APT campaign, is installed on a network machine and the malicious traffic is recorded. Different classification algorithms are trained on the collected traffic to build the detection model. The authors in [33] employ the Data Leakage Prevention (DLP) for APT detection. The proposed methodology is based on monitoring the data traffic and utilise the DLP to detect data leaks. Next, leaks patterns are created based on the attributes of the data leaks.

From the presented related work, we can see that APT forms a problem for the current IDS in terms of real time detections. Furthermore, the steps of APT may occur over a day, week, month or even a year. This makes the alerts correlation a serious challenge. Additionally, missing the detection of one step or more of APT campaign makes it difficult to investigate the full APT scenario. A correlation-based system is proposed in [10] for APT detection, and this work is a step towards building the proposed framework.

4 DISGUISED EXECUTABLE FILE DETECTION (DEFD)

According to Mandiant APT1 report [34], spear phishing is the most commonly used technique to get the point of entry in APT. The spear phishing emails contain either a malicious attachment or a hyperlink to a malicious file. The subject line and the text in the email body are usually relevant to the recipient. Executable files supposed to end in .exe are made to appear as simple document files (pdf, doc, ppt, excel) to convince the victim to click on it.

The DeFD module detects disguised exe files transferred over the connections. Meaning, it detects if the content of the file is exe while the extension is not exe. Figure 2 shows the methodology of DeFD; the network traffic is processed, all connections are analysed and all exe files identified when transferring over the connections are filtered. This filtering is based on the file content. Following this, the file name extension should be checked to decide about raising an alert on disguised exe file detection.

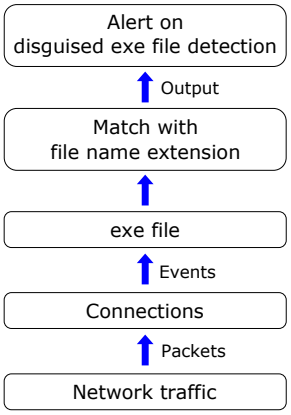

Figure 2: Methodology of the disguised exe file detection.

Different security systems are used to monitor the network traffic [35]. *Bro* is used to implement the DeFD module. *Bro* is a network security monitor which can passively analyse the network traffic and generate informative events. These events provide information about specific network activities [36].

Algorithm 1 shows the implementation pseudo-code of the DeFD module. The network traffic is processed; this module waits for *file_over_new_connection* event to be generated by Bro. This event indicates that a file has been seen in the process of being transferred over a connection [37]. Then *describe()* function is applied on that file; this function provides a text description regarding metadata of the file, so the file name can be extracted.

This method is able to detect disguised exe files in both cases, uploaded and downloaded, but DeFD aims to detect only downloaded disguised exe files, as they are the ones used in the second step of APT. Thus, DeFD checks the current connection, in which a new file has seen being transferred, if it is established by a host from the monitored network. This is done by checking the connection source IP address through *is_local_addr* function; this function returns true if an address corresponds to one of the defined local networks, false if not. For this reason, the *subnet* of the monitored network should be defined.

Following this, the *mime_type* [38], which can be extracted from *file_over_new_connection* event, is checked for its presence in *t_exe_file* table. *t_exe_file* table contains the MIME types of the files which DeFD aims to find and filter, i.e. MIME types of exe files. Therefore, if the transferred file is an exe file (based on its *mime_type*), DeFD checks whether the extension in the file name

#fields	timestamp	alert_type	orig_h	orig_p	resp_h	resp_p	infected_host	malicious_file
#types	time	string	addr	port	addr	port	addr	string
1407424021.202210		disguised_exe_alert	[redacted]	56973	207.244.73.42	80	[redacted]	SkypeSetup.pdf
1407424040.255414		disguised_exe_alert	[redacted]	53105	207.244.73.42	80	[redacted]	ViberSetup.doc
#close	2014-08-07-19-15-07							

Figure 3: The log produced by the DeFD module.

Algorithm 1 Implementation pseudo-code of DeFD

```
Input: t_exe_file table
Get file_over_new_connection event
fname ← file name
if the connection is established by one of the network's hosts then
    if the file MIME type is in t_exe_file table then
        if file MIME type = fname extension then
            if the same disguised_exe_alert has been raised during the previous 24 hours then
                goto End
            else
                Generate an event (disguised_exe_alert)
                Write disguised_exe_alert into disguised_exe_detection.log
                Notify the network security team via email
                Deny generating the same disguised_exe_alert during the next 24 hours
            end if
        else
            goto End
        end if
    else
        goto End
    end if
else
    goto End
end if
End
```

is exe; this file name extension is extracted from the output of the *describe()* function previously mentioned. If the file name extension is not exe, this means it is a disguised exe file. Before an alert is raised, DeFD checks if an alert regarding the same host and for the same disguised exe file has been generated during the previous day. This check is to ensure that DeFD does not generate the same alert about the same set (host, file) during one day, therefore, DeFD checks if the current set exists in the *t_suppress_disguised_exe_alert* table, this table contains all detected sets during the last day.

If the current set (host, file) had not been detected during the previous 24 hours, DeFD generates *disguised_exe_alert* event to be used in the FCI correlation framework. The malicious connection information is written into a specific log *disguised_exe_detection.log*, to keep a historical record of the monitored network. An alert email regarding disguised exe file detection is sent to RT (Request Tracker) [39], where the network security team can perform additional forensics and respond to it. The current detected set (host, file) is added into the *t_suppress_disguised_exe_alert* table where it stays for one day to ensure that DeFD does not generate another alert about the same set during the same day. The written information into *disguised_exe_detection.log* is:

```
timestamp = c$start_time,
alert_type = "disguised_exe_alert"
connection  = c$id
infected_host = c$id$orig_h
malicious_file = fname
```

5 EVALUATION RESULTS

To evaluate the effectiveness of the DeFD module, a download of disguised exe file was simulated via the campus network. An experimental server was set up, using Bro, to passively monitor the campus live traffic. DeFD was run on that experimental server for the purpose of disguised exe file detection. Two exe files were randomly selected, *SkypeSetup.exe* and *ViberSetup.exe*, and their names' extensions (*exe*) were changed into *pdf* and *doc* extensions, i.e. *SkypeSetup.pdf* and *ViberSetup.doc* respectively. The two disguised exe files were uploaded to the *speedyshare.com* public server. Then a host working on a computer connected to the Internet through the monitored network was used to connect to that public server and download the modified files. As shown in Figure 3, DeFD was able to detect both malicious downloads and write the information regarding each connection into a specific log.

This experiment was repeated a hundred of times using different disguised exe files with different extensions. DeFD was able to detect all the malicious files, the average detection delay was 270 ms with a standard deviation of 54 ms. Figure 4 shows an example of a *Disguised_exe* ticket, which was emailed to RT.

Greetings,

the security team CSIRT-MU detected involvement of the IP address [redacted] into the following incident:

Incident type: Disguised_exe
Time of detection: 2014-08-10 15:07:01 +0100
IP address [redacted]
Domain name: ---

Details of this incident can be found at this address:
https://reports.csirt.muni.cz/A4FE72DC-65A8-1C74-8627-5664BE78D732

Best regards,
CSIRT-MU, the security team

Date: Thu, 10 Aug 2014 15:07:19 +0100

Figure 4: An example of a *Disguised_exe* ticket.

6 CONCLUSION AND FUTURE WORK

The appearance of new forms of cyber attacks, such as APT, has created new challenges to the current (IDSs). APT has emerged as a serious threat to the cyber security hitting selected companies and organisations. This paper presents the disguised exe file detection (DeFD) module, which aims at detecting disguised exe files transferred over the network connections. The detection is based on a comparison between the *MIME type* of the transferred file and the file name extension.

As our future work, this module is to be used as part of a correlation-based IDS system for APT detection. The APT step detected by this module should be correlated with other steps, which are detected by other modules, to find the full APT scenario.

REFERENCES

[1] The-Sun. Nhs cyber attack. https://www.thesun.co.uk/topic/nhs-cyber-attack/. Accessed: 01-09-2017.

[2] The-Guardian. Petya' ransomware attack: what is it and how can it be stopped? https://www.theguardian.com/technology/2017/jun/27/petya-ransomware-cyber-attack-who-what-why-how. Accessed: 01-09-2017.

[3] Steve Morgan. Hackerpocalypse: A cybercrime revelation. *2016 Cybercrime Report, Cybersecurity Ventures*, 2016.

[4] Nir Nissim, Aviad Cohen, Chanan Glezer, and Yuval Elovici. Detection of malicious pdf files and directions for enhancements: a state-of-the art survey. *Computers & Security*, 48:246–266, 2015.

[5] Ibrahim Ghafir and Vaclav Prenosil. Advanced persistent threat attack detection: An overview. *International Journal of Advances in Computer Networks and Its Security (IJCNS)*, vol. 4(Issue 4):50–54, 2014. ISSN 2250-3757.

[6] Paul Wood, Mathew Nisbet, Gerry Egan, Nicholas Johnston, Kevin Haley, Bhaskar Krishnappa, Tuan-Khanh Tran, Irfan Asrar, Orla Cox, Sean Hittel, et al. Symantec internet security threat report trends for 2011. *Volume XVII*, 2012.

[7] Ibrahim Ghafir and Vaclav Prenosil. Advanced persistent threat and spear phishing emails. In *Proceedings of International Conference on Distance Learning, Simulation and Communication*, pages 34–41. University of Defence, 2015.

[8] Ibrahim Ghafir, Jibran Saleem, Mohammad Hammoudeh, Hanan Faour, Vaclav Prenosil, Sardar Jaf, Sohail Jabbar, and Thar Baker. Security threats to critical infrastructure: the human factor. *The Journal of Supercomputing*, pages 1–17, 2018.

[9] Tatwadarshi P Nagarhalli, JW Bakal, and Neha Jain. A brief survey of detection and mitigation techniques for clickjacking and drive-by download attacksfi... *International Journal of Computer Applications (0975–8887)*, 138(2), 2016.

[10] Ibrahim Ghafir and Vaclav Prenosil. Proposed approach for targeted attacks detection. In *Advanced Computer and Communication Engineering Technology*, pages 73–80. Springer, 2016.

[11] Ibrahim Ghafir, Jakub Svoboda, and Vaclav Prenosil. Tor-based malware and tor connection detection. In *International Conference on Frontiers of Communications, Networks and Applications (ICFCNA)*, pages 1–6. IEEE Xplore Digital Library, 2014.

[12] Ibrahim Ghafir, Vaclav Prenosil, Mohammad Hammoudeh, Liangxiu Han, and Umar Raza. Malicious ssl certificate detection: A step towards advanced persistent threat defence. In *Proceedings of International Conference on Future Networks and Distributed Systems*. ACM Digital Library, Cambridge, United Kingdom, 2017.

[13] Ibrahim Ghafir and Vaclav Prenosil. Malicious file hash detection and drive-by download attacks. In *Proceedings of the Second International Conference on Computer and Communication Technologies*, pages 661–669. Springer, 2016.

[14] Ibrahim Ghafir and Vaclav Prenosil. DNS traffic analysis for malicious domains detection. In *2nd International Conference on Signal Processing and Integrated Networks (SPIN)*, pages 613–918. IEEE Xplore Digital Library, 2015.

[15] Ibrahim Ghafir and Vaclav Prenosil. DNS query failure and algorithmically generated domain-flux detection. In *International Conference on Frontiers of Communications, Networks and Applications (ICFCNA)*, pages 1–5. IEEE Xplore Digital Library, 2014.

[16] Ibrahim Ghafir and Vaclav Prenosil. Blacklist-based malicious ip traffic detection. In *Global Conference on Communication Technologies (GCCT)*, pages 229–233. IEEE Xplore Digital Library, 2015.

[17] Trend Micro white paper. The custom defense against targeted attacks. http://www.trendmicro.fr/media/wp/custom-defense-against-targeted-attacks-whitepaper-en.pdf. Accessed: 10-11-2017.

[18] Ibrahim Ghafir, Vaclav Prenosil, Ahmad Alhejailan, and Mohammad Hammoudeh. Social engineering attack strategies and defence approaches. In *Future Internet of Things and Cloud (FiCloud), 2016 IEEE 4th International Conference on*, pages 145–149. IEEE, 2016.

[19] Ibrahim Ghafir, Jakub Svoboda, and Vaclav Prenosil. A survey on botnet command and control traffic detection. *International Journal of Advances in Computer Networks and its security (ICJNS)*, vol. 5(Issue 2):75–80, 2015.

[20] Ibrahim Ghafir, Vaclav Prenosil, and Mohammad Hammoudeh. Botnet command and control traffic detection challenges: A correlation-based solution. *International Journal of Advances in Computer Networks and Its Security (IJCNS)*, vol. 7 (Issue 2):27–31, 2015.

[21] Sandeep Yadav, Ashwath Kumar Krishna Reddy, AL Narasimha Reddy, and Supranamaya Ranjan. Detecting algorithmically generated domain-flux attacks with dns traffic analysis. *Networking, IEEE/ACM Transactions on*, 20(5):1663–1677, 2012.

[22] Klaas Apostol. Brute-force attack. 2012.

[23] Bashar Ewaida. Pass-the-hash attacks: Tools and mitigation. *Last accessed September*, 11, 2013.

[24] Chirag Modi, Dhiren Patel, Bhavesh Borisaniya, Hiren Patel, Avi Patel, and Muttukrishnan Rajarajan. A survey of intrusion detection techniques in cloud. *Journal of Network and Computer Applications*, 36(1):42–57, 2013.

[25] Rob Jansen, Florian Tschorsch, Aaron Johnson, and Björn Scheuermann. The sniper attack: Anonymously deanonymizing and disabling the tor network. Technical report, DTIC Document, 2014.

[26] Xu Wang, Kangfeng Zheng, Xinxin Niu, Bin Wu, and Chunhua Wu. Detection of command and control in advanced persistent threat based on independent access. In *Communications (ICC), 2016 IEEE International Conference on*, pages 1–6. IEEE, 2016.

[27] Paul Giura and Wei Wang. A context-based detection framework for advanced persistent threats. In *Cyber Security (CyberSecurity), 2012 International Conference on*, pages 69–74. IEEE, 2012.

[28] Guillaume Brogi and Valérie Viet Triem Tong. Terminaptor: Highlighting advanced persistent threats through information flow tracking. In *New Technologies, Mobility and Security (NTMS), 2016 8th IFIP International Conference on*, pages 1–5. IEEE, 2016.

[29] Joseph Sexton, Curtis Storlie, and Joshua Neil. Attack chain detection. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 8(5-6):353–363, 2015.

[30] Boldizsár Bencsáth, Gábor Pék, Levente Buttyán, and Márk Félegyházi. Duqu: Analysis, detection, and lessons learned. In *ACM European Workshop on System Security (EuroSec)*, volume 2012, 2012.

[31] J Vijaya Chandra, Narasimham Challa, and Sai Kiran Pasupuleti. A practical approach to e-mail spam filters to protect data from advanced persistent threat. In *Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on*, pages 1–5. IEEE, 2016.

[32] Saranya Chandran, P Hrudya, and Prabaharan Poornachandran. An efficient classification model for detecting advanced persistent threat. In *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*, pages 2001–2009. IEEE, 2015.

[33] Johan Sigholm and Martin Bang. Towards offensive cyber counterintelligence: Adopting a target-centric view on advanced persistent threats. In *Intelligence and Security Informatics Conference (EISIC), 2013 European*, pages 166–171. IEEE, 2013.

[34] FireEye. Mandiant apt1 report. https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/mandiant-apt1-report.pdf. Accessed: 26-06-2016.

[35] Jakub Svoboda, Ibrahim Ghafir, and Vaclav Prenosil. Network monitoring approaches: An overview. *International Journal of Advances in Computer Networks and Its Security (IJCNS)*, vol. 5(Issue I), 2015.

[36] Vern Paxson. Bro: a system for detecting network intruders in real-time. *Computer networks*, 31(23):2435–2463, 1999.

[37] Bro Project. file_over_new_connection event. https://www.bro.org/sphinx/scripts/base/bif/event.bif.bro.html#id-file_new. Accessed: 01-06-2016.

[38] MrForms. Mime types list. http://www.freeformatter.com/mime-types-list.html. Accessed: 07-04-2015.

[39] Best-Practical-Solutions. Rt: Request tracker. https://www.bestpractical.com/rt/. Accessed: 15-02-2017.